



UNITED STATES PATENT AND
Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
-----------------	-------------	----------------------	---------------------

09/226,939 01/08/99 VINCENT J 346872000500

WM02/1218

MARTIN C. FLIESLER, ESQ.
FLIESLER, DUBB, MEYER & LOVEJOY LLP
FOUR EMBARCADERO CENTER
SUITE 400
SAN FRANCISCO CA 94111-4156

EXAMINER

L.Y.A

ART UNIT

PAPER NUMBER

2172

DATE MAILED:

12/18/00

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

Office Action Summary	Application No. 09/226,939	Applicant(s) Vincent et al.
	Examiner Anh Ly	Group Art Unit 2172

- Responsive to communication(s) filed on _____.
- This action is **FINAL**.
- Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11; 453 O.G. 213.

A shortened statutory period for response to this action is set to expire 3 month(s), or thirty days, whichever is longer, from the mailing date of this communication. Failure to respond within the period for response will cause the application to become abandoned. (35 U.S.C. § 133). Extensions of time may be obtained under the provisions of 37 CFR 1.136(a).

Disposition of Claims

- Claim(s) 1-30 is/are pending in the application.
- Of the above, claim(s) _____ is/are withdrawn from consideration.
- Claim(s) _____ is/are allowed.
- Claim(s) 1-30 is/are rejected.
- Claim(s) _____ is/are objected to.
- Claims _____ are subject to restriction or election requirement.

Application Papers

- See the attached Notice of Draftsperson's Patent Drawing Review, PTO-948.
- The drawing(s) filed on _____ is/are objected to by the Examiner.
- The proposed drawing correction, filed on _____ is approved disapproved.
- The specification is objected to by the Examiner.
- The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

- Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).
- All Some* None of the CERTIFIED copies of the priority documents have been
- received.
 - received in Application No. (Series Code/Serial Number) _____.
 - received in this national stage application from the International Bureau (PCT Rule 17.2(a)).
- *Certified copies not received: _____.
- Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

- Notice of References Cited, PTO-892
- Information Disclosure Statement(s), PTO-1449, Paper No(s). 2 & 4
- Interview Summary, PTO-413
- Notice of Draftsperson's Patent Drawing Review, PTO-948
- Notice of Informal Patent Application, PTO-152

--- SEE OFFICE ACTION ON THE FOLLOWING PAGES ---

Art Unit: 2172

DETAILED ACTION

1. Claims 1-30 are pending in this application.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103© and potential 35 U.S.C. 102(f) or (g) prior art under 35 U.S.C. 103(a).

3. Claims 1-16, and 22-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No. 6,108,659 issued to Vincent in view of US Patent No. 5,832,484 issued to Sankaran et al. (“Sankaran”).

Art Unit: 2172

With respect to claim 1, Vincent discloses a method of automatically generating a database code object, the method comprising the step of applying a recursive algorithm that queries a database; representing dependencies that do not involve dependencies on triggers and on implementations of object-oriented code objects in the database; applying the recursive algorithm on each of the object-oriented code objects; using the recursive algorithm on each of the triggers to incorporate; and repeating the procedure for incorporating dependencies of implementation of object-oriented code objects as claimed (see abstract, col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67, and col. 8, lines 35-44).

Vincent does not explicitly indicate "dependency graph of a database code object, and the direct dependency graph containing dependencies."

However, Sankaran discloses the dependency graph and directed dependency graph (col. 18, lines 4-16, and lines 37-59, col. 20, lines 10-41).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

Art Unit: 2172

With respect to claim 2, Vincent discloses a database code object debugging tool (col. 1, lines 13-30).

With respect to claim 3, Vincent discloses a database code coverage tool (col. 1, lines 29-30).

With respect to claim 4, Vincent discloses a database code object profiling tool (col. 1, lines 13-30).

With respect to claim 5, Vincent discloses a database code object testing tool (col. 1, lines 29-30, and lines 55-67, and col. 2, lines 1-8).

With respect to claim 6, Vincent discloses a database code objects that are invalid set (col. 4, lines 17-47).

With respect to claim 7, Vincent discloses a database code object (col. 2, lines 40-57).

With respect to claim 8, Vincent discloses a tool for development (col. 1, lines 13-30).

With respect to claim 9, Vincent discloses a method of generating a code object, the triggers, and on implementations of object-oriented code objects, the method comprising the steps of querying a database catalog; doing the query recursively as claimed (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29).

Vincent does not explicitly indicate “dependency tree or data structure.”

However, Sankaran discloses the dependency tree (col. 67, lines 34-67, and col. 68, lines 1-65).

Art Unit: 2172

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

With respect to claim 10, Vincent discloses a database code object debugging tool (col. 1, lines 13-30).

With respect to claim 11, Vincent discloses a database code coverage tool (col. 1, lines 29-30).

With respect to claim 12, Vincent discloses a database code object profiling tool (col. 1, lines 13-30).

With respect to claim 13, Vincent discloses a database code object testing tool (col. 1, lines 29-30, and lines 55-67, and col. 2, lines 1-8).

With respect to claim 14, Vincent discloses a database code objects that are invalid set (col. 4, lines 17-47).

With respect to claim 15, Vincent discloses a database code object (col. 2, lines 40-57).

With respect to claim 16, Vincent discloses a tool for development (col. 1, lines 13-30).

Art Unit: 2172

With respect to claim 22, Vincent discloses a method of generating a code object, the implementations of object-oriented code objects in a database, the method comprising the steps of querying a database catalog; doing the query recursively as claimed (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67, and col. 8, 35-44).

Vincent does not explicitly indicate "dependency graph."

However, Sankaran discloses the dependency graph (col. 67, lines 34-67, and col. 68, lines 1-65).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

With respect to claim 23, Vincent discloses a database code coverage tool (col. 1, lines 29-30).

With respect to claim 24, Vincent discloses a database code object profiling tool (col. 1, lines 13-30).

Art Unit: 2172

With respect to claim 25, Vincent discloses a database code object testing tool (col. 1, lines 29-30, and lines 55-67, and col. 2, lines 1-8).

With respect to claim 26, Vincent discloses a database code objects that are invalid set (col. 4, lines 17-47).

With respect to claim 27, Vincent discloses a system for identifying a target data base code object. The system comprising the implementations of packages, specifications of types, implementations of types and triggers, data structure (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29).

Vincent does not explicitly indicate "dependency graph."

However, Sankaran discloses the dependency graph (col. 67, lines 34-67, and col. 68, lines 1-65).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

Art Unit: 2172

With respect to claim 28, Vincent discloses a method for generating a target data base code object, the method comprising the steps of the data comprising representations of object-oriented code objects, specifications of packages; using a recursive code mechanism, triggers (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67).

Vincent does not explicitly indicate "dependency graph."

However, Sankaran discloses the dependency graph (col. 67, lines 34-67, and col. 68, lines 1-65).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

With respect to claim 29, Vincent discloses a computer program product embedded on a computer readable medium for in debugging a target data base code object comprising a recursive code mechanism; having entries to contain representations of code objects, specifications of

Art Unit: 2172

packages, triggers (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67).

Vincent does not explicitly indicate "dependency graph."

However, Sankaran discloses the dependency graph (col. 67, lines 34-67, and col. 68, lines 1-65).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

With respect to claim 30, Vincent discloses a computer program product embedded on a computer readable medium for in debugging a target data base code object comprising a program code mechanism for applying recursive algorithm that queries a database; on each of the object-oriented code objects; for paring the source code of the code object; for applying the recursive algorithm on each of the triggers; for repeating the procedure for incorporating; and for

Art Unit: 2172

debugging as claimed (col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67).

Vincent does not explicitly indicate “dependency graph.”

However, Sankaran discloses the dependency graph (col. 67, lines 34-67, and col. 68, lines 1-65).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of Sankaran so as to have a complete dependency graph of a database code object because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

4. Claims 17-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No. 6,108,659 issued to Vincent in view of US Patent No. 5,325,531 issued to McKeeman et al. (“McKeeman”).

With respect to claim 17, Vincent discloses a method of automatically generating a code object on database triggers, the method comprising the step of applying a recursive algorithm that queries a database; representing dependencies that do not involve dependencies on triggers and on

Art Unit: 2172

implementations of object-oriented code objects in the database; applying the recursive algorithm on each of the object-oriented code objects; using the recursive algorithm on each of the triggers to incorporate; and repeating the procedure for incorporating dependencies of implementation of object-oriented code objects as claimed (see abstract, col. 3, lines 40-67, col. 4, lines 65-67, col. 5, lines 1-29, col. 6, lines 12-23, col. 7, lines 20-67, and col. 8, lines 35-44).

Vincent does not explicitly indicate "the method of generating dependency information including dependencies."

However, McKeeman discloses the dependency information (col. 6, lines 22-67, col. 17, lines 47-67, and col. 18, lines 1-3).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Vincent with the teachings of McKeeman so as to have a method of generating dependency information including dependencies of code objects on database triggers because the combination would eliminate coding errors while declaring parameter data types in the anonymous blocks, at the same time initializing values for the declared parameters; allow for executing subprograms that take complex user defined types as parameters and also provide a complete execution facility that can be used to execute a code object (Vincent - col. 2, lines 9-37) in the database code object debugging environment.

With respect to claim 18, Vincent discloses a database code coverage tool (col. 1, lines 29-30).

Art Unit: 2172

With respect to claim 19, Vincent discloses a database code object profiling tool (col. 1, lines 13-30).

With respect to claim 20, Vincent discloses a database code object testing tool (col. 1, lines 29-30, and lines 55-67, and col. 2, lines 1-8).

With respect to claim 21, Vincent discloses a database code objects that are invalid set (col. 4, lines 17-47).

Conclusions

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosures.

Schmidt et al. (US Patent No. 4,558,413)

Drewry et al. (US. Patent No. 5,925,100)

Wrabetz et al. (US. Patent No. 5,442,791)

Mitchell et al. (US. Patent No. 5,872,973)

Jordan (US. Patent No. 6,094,528)

McKeeman et al. (US Patent No. 5,170,465)

Contact Information

Art Unit: 2172

6. Any inquiry concerning this communication should be directed to Anh Ly whose telephone number is (703) 306-4527. The examiner can be reached on Monday - Friday from 8:00 AM to 4:00 PM.

If attempts to reach the examiner are unsuccessful, see the examiner's supervisor, Kim Vu, can be reached on (703) 305-4393.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 308-9051 (for formal communications intended for entry)

or:

(703) 305-9724 or (703) 308-6606 (for informal or draft communications, please label
"PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive,
Arlington, VA, Sixth Floor (receptionist).

Inquiries of a general nature or relating to the status of this application should be directed
to the Group receptionist whose telephone number is (703) 305-9600.

AL

Dec. 5th, 2000


KIM VU
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100